DCS 1202:
OBJECT ORIENTED PROGRAMMING

JAVA

Peter Mackenzie petnjau@gmail.com

CHAPTER ONE:

INTRODUCTION TO OBJECT ORIENTED PROGRAMMING CONCEPTS

OBJECT ORIENTED PROGRAMMING

What is OOP?

- **Object-oriented programming (OOP) is an engineering approach for building software systems**
  - **Based on the concepts of classes and objects that are used for modeling the real world entities**
- **Object-oriented programs**
  - **Consist of a group of cooperating objects**
  - **Objects exchange messages, for the purpose of achieving a common objective**
  - **Implemented in object-oriented languages**

CONT…..

- A program models a world of interacting objects
- Objects create other objects and "send messages" to each other (in Java, call each other's methods)
- Each object belongs to a class
  - A class defines properties of its objects
  - The data type of an object is its class
- Programmers write classes (and reuse existing classes)

EXAMPLES OF OOP LANGUAGES

- **All modern languages are object-oriented:**
  - **Java, C#, PHP, Perl, C++,**

OOP CONCEPTS/ PRINCIPLES

1. Class
2. Object
3. Inheritance
4. Polymorphism
5. Abstraction
6. Encapsulation
7. Messages

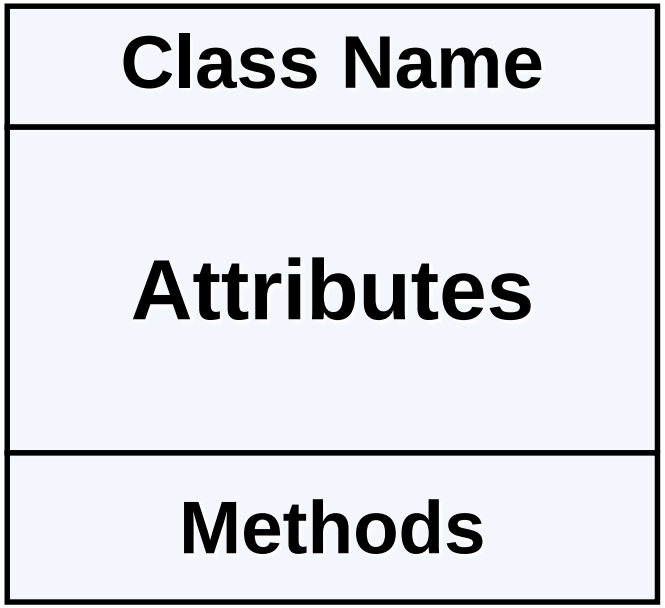
1. CLASS

- Class is a blueprint of data and functions or methods.
- A class is a blueprint for any functional entity which defines its properties and functions.
- Classes provide the structure for *objects*
- Classes define:
  - Set of ***attribute*** also called *state* Represented by variables and properties
  - Behavior Represented by methods
- A class defines the methods and types of data associated with an object

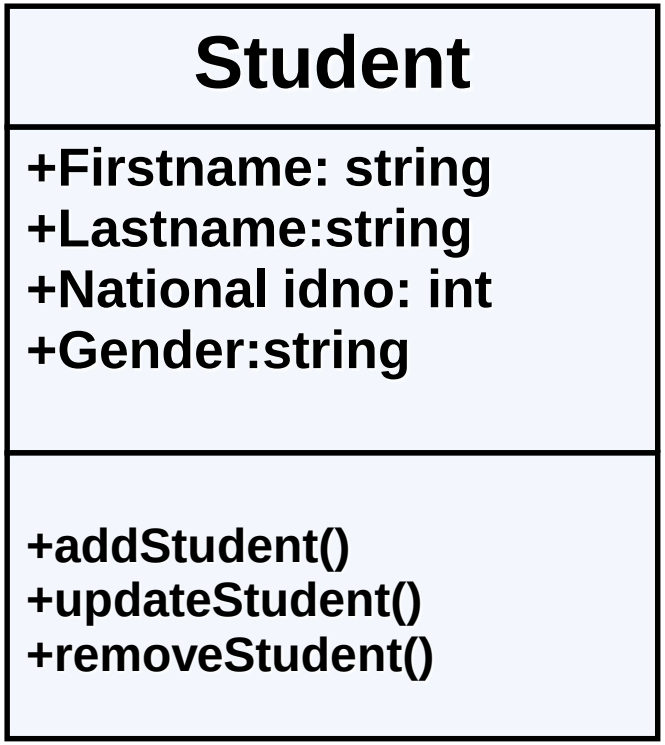
COMPONENTS OF A CLASS

- A class has three components:
  1. **Class name**
     - It's the class identification/it's the name of the entity
  2. **Attributes**
     - These are the characteristics exhibited by the objects/ defines the state.
  3. **Methods/functions**
     - These defines the behavior of the objects of the class

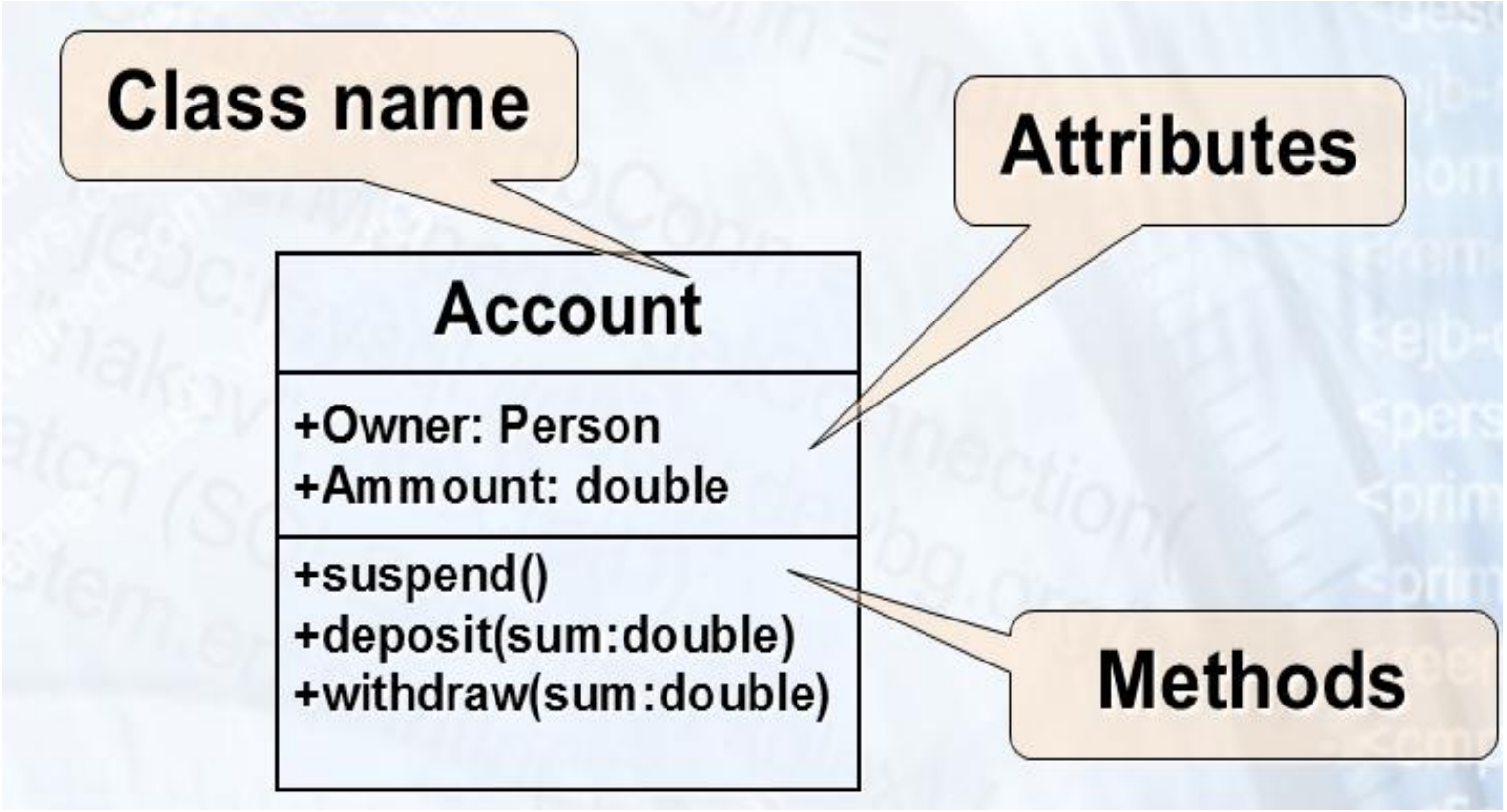
CONT…..



EXAMPLE OF A CLASS



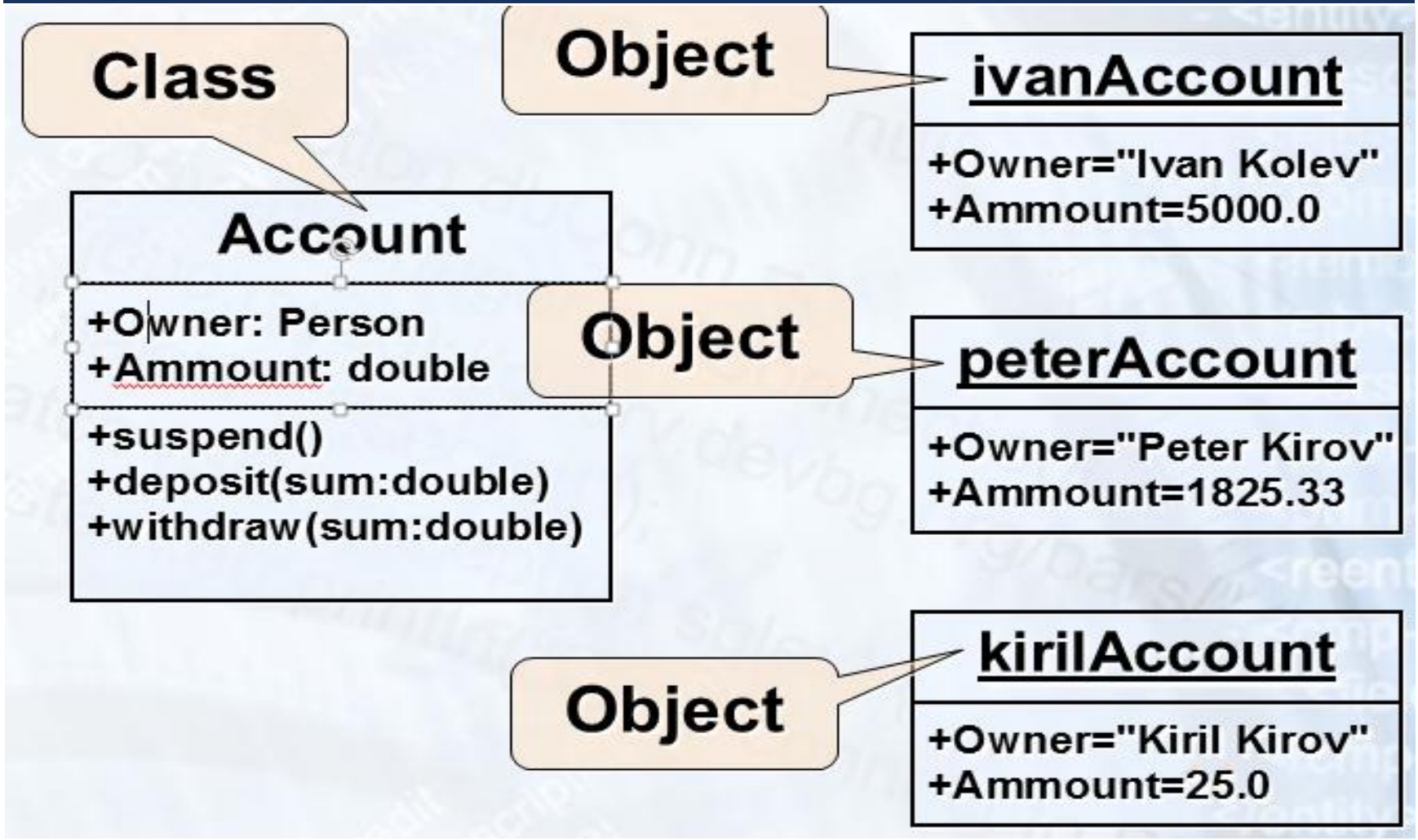
EXAMPLE OF A CLASS



2. OBJECT

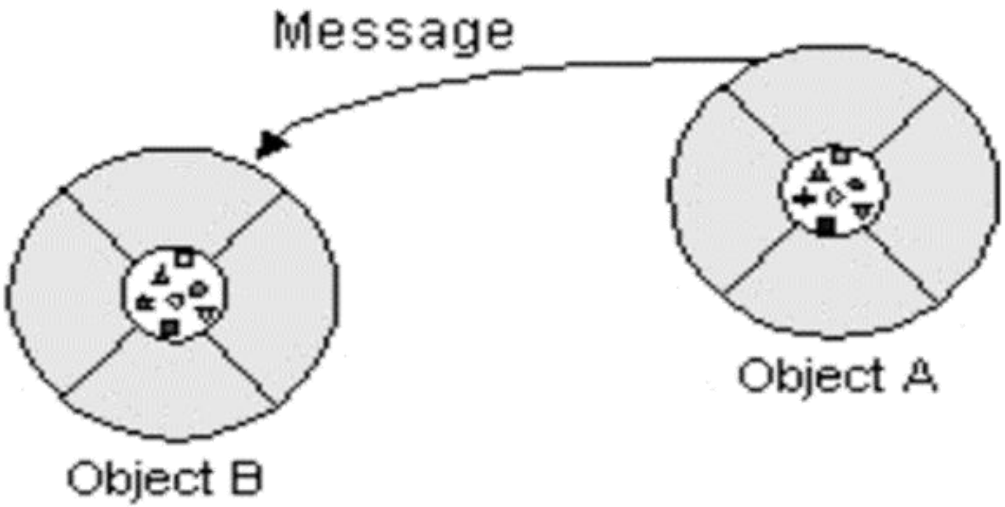
- An *object* is a concrete ***instance*** of a particular class
- Creating an object from a class is called ***instantiation***
- **Object** means a real word entity such as pen, chair, table etc.
- **Objects have state**
  - Set of values associated to their attributes
- **Example:**
  - **Class**: Account
  - **Objects**: Ivan's account, Peter's account

CLASSES AND OBJECTS – EXAMPLE



3.MESSAGES

- A request for an object to perform one of its operations (methods)
- All communication between objects is done via messages



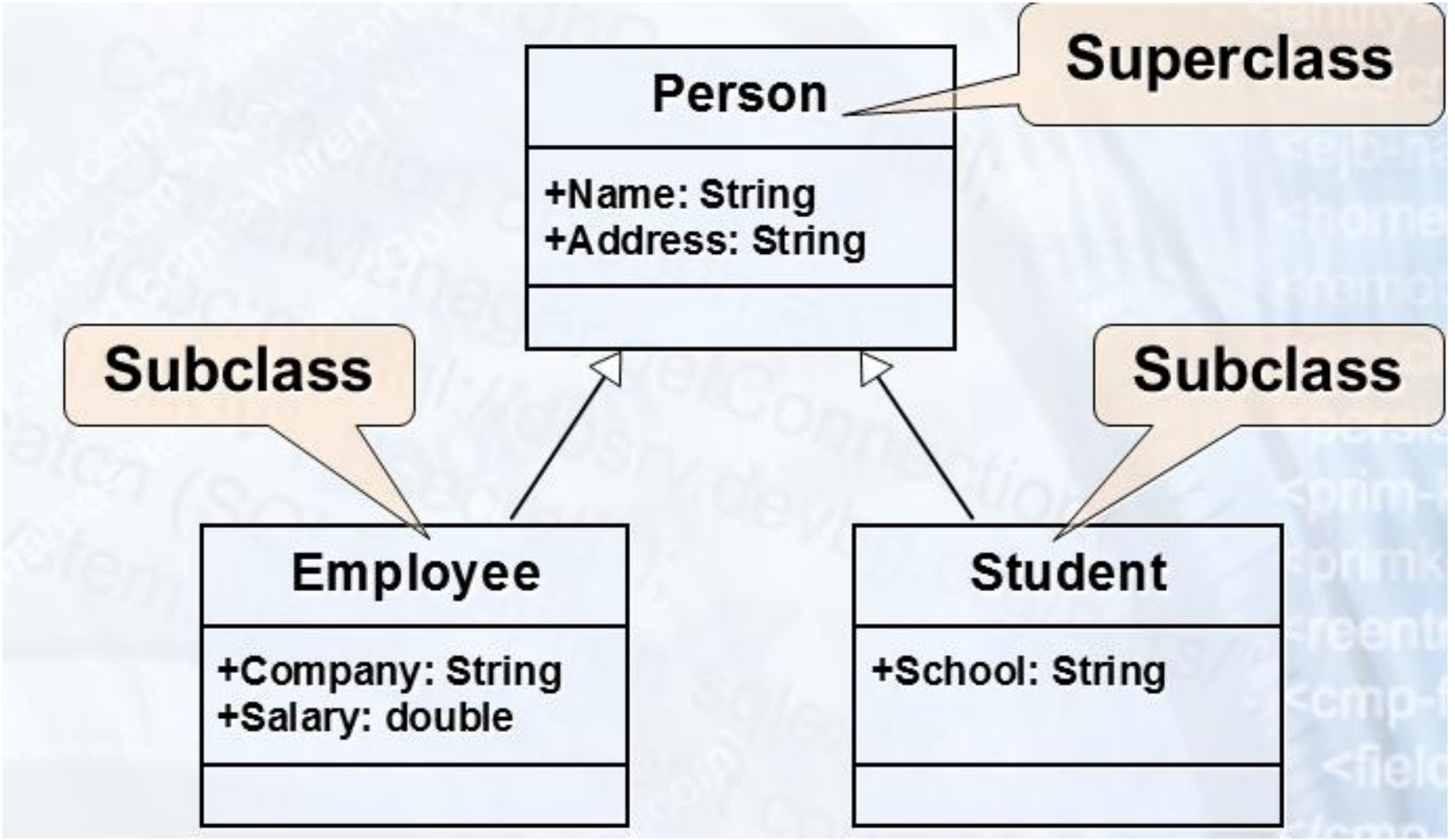
4. INHERITANCE

- inheritance is the process by which objects of one class acquire the properties of objects of another class.
- A class can *extend* another class, inheriting all its data members and methods
  - The child class can redefine some of the parent class's members and methods and/or add its own
- A class can *implement* an interface, implementing all the specified methods
- Inheritance implements the "is a" relationship between objects

CONT……

- Inheritance provides re usability.
  - *This means that we can add additional features to an existing class without modifying it.*

CONT.....


Person
+Name: String
+Address: String
Superclass
Subclass
Subclass
Employee
+Company: String
+Salary: double
Student
+School: String

CONT.....

- **In Java, a subclass can extend only one superclass**
- **In Java, a sub interface can extend one super interface**
- **In Java, a class can implement several interfaces**
  - **This is Java's form of *multiple inheritance***

BENEFITS OF INHERITANCE

- **Inheritance plays a dual role:**
  - **A subclass reuses the code from the superclass**
  - **A subclass inherits the *data type* of the superclass (or interface) as its own secondary type**

5. POLYMORPHISM

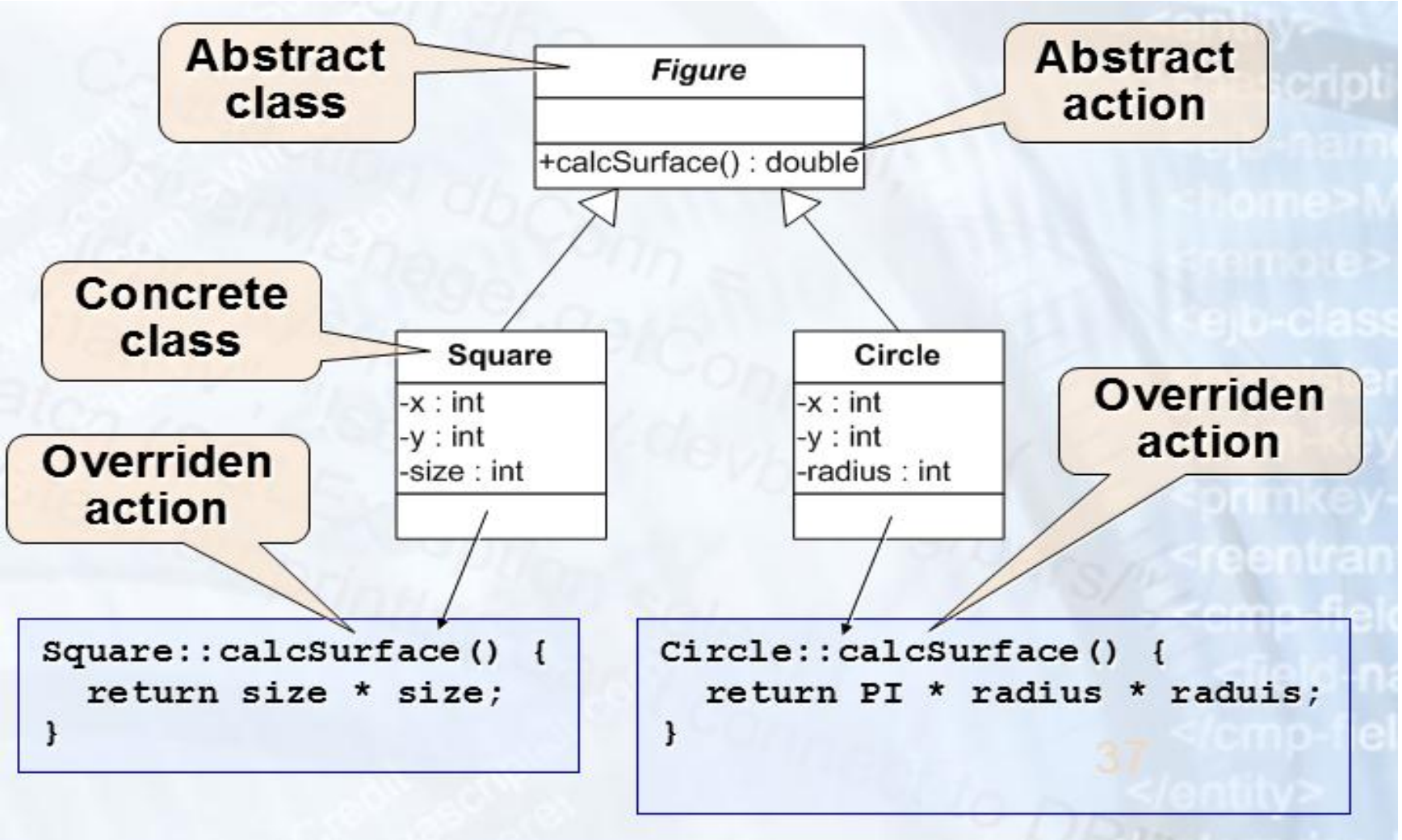
- Polymorphism means ability to take more than one form.
- An operation may exhibit different behaviors in different instances.
- The behavior depends upon the types of data used in the operation. polymorphism is extensively used in implementing inheritance.
- Polymorphism is a concept, which allows us to redefine the way something works, by either changing how it is done or by changing the parts using which it is done. Both the ways have different terms for them.

CONT…

- **Polymorphism Ability to take more than one form**
  - **A class can be used through its parent class's interface**
  - **A subclass may override the implementation of an operation it inherits from a superclass (late binding)**
- **Polymorphism allows abstract operations to be defined and used**
  - **Abstract operations are defined in the base class's interface and implemented in the subclasses**

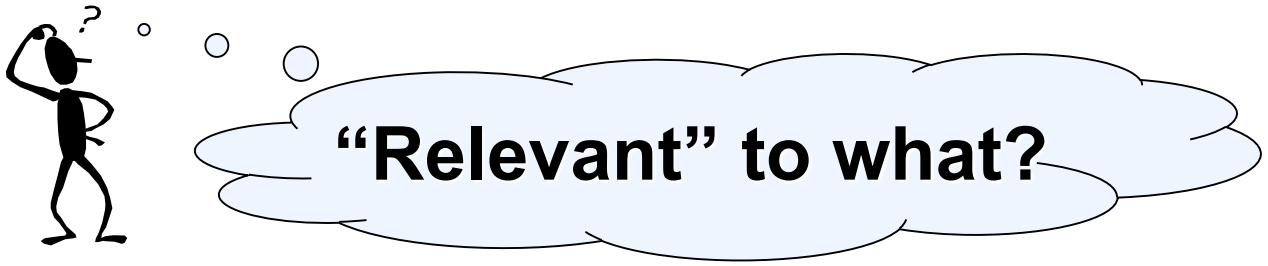
EXAMPLE POLYMORPHISM: "SPEAK SOMETHING"

POLYMORPHISM – EXAMPLE



6. ABSTRACTION

- **Abstraction means ignoring irrelevant features, properties, or functions and emphasizing the relevant ones...**



- **... relevant to the given project (with an eye to future reuse in similar projects)**
- **Abstraction = managing complexity**

CONT.....

- Abstraction means, showcasing only the required things to the outside world while hiding the details.
- Continuing our example, **Human Being's** can talk, walk, hear, eat, but the details are hidden from the outside world.
- We can take our skin as the Abstraction factor in our case, hiding the inside mechanism.

CONT…

- Abstraction is something we do every day
  - Looking at an object, we see those things about it that have meaning to us
  - We abstract the properties of the object, and keep only what we need
- Allows us to represent a complex reality in terms of a simplified model
- Abstraction highlights the properties of an entity that we are most interested in and hides the others

7. ENCAPSULATION

- Wrapping up(combing) of data and functions into a single unit is known as encapsulation.
- The data is not accessible to the outside world and only those functions which are wrapping in the class can access it.
- This insulation of the data from direct access by the program is called **data hiding or information hiding**.
- It can also be said data binding. Encapsulation is all about binding the data variables and functions together in class.

CONT.…

- **Encapsulation means that all data members (*fields*) of a class are declared *private***
  - **Some methods may be private, too**
- **The class interacts with other classes (called the *clients* of this class) only through the class's constructors and public methods**
- **Constructors and public methods of a class serve as the *interface* to class's clients**

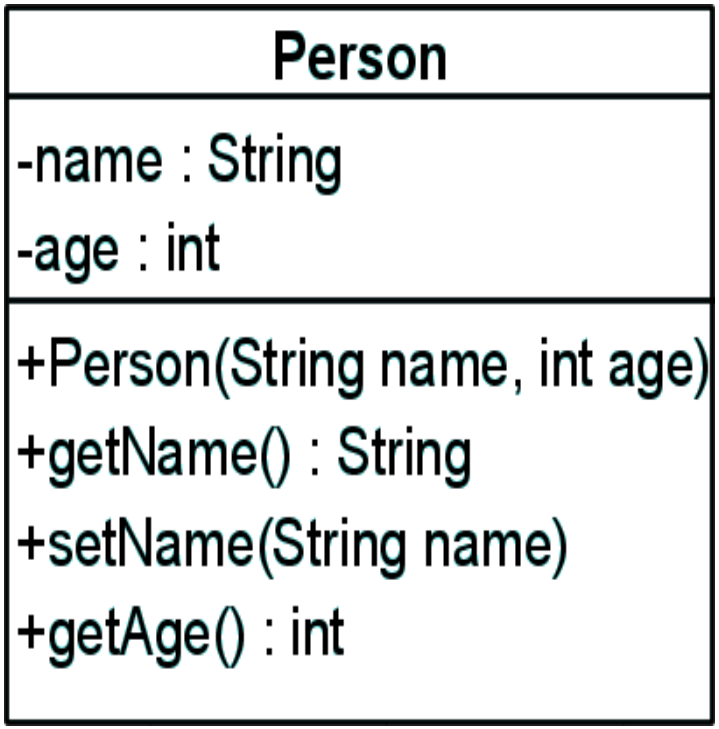
CONT.....

- For example: **capsule**, it is wrapped with different medicines.
  - A java class is the example of encapsulation. Java bean is the fully encapsulated class because all the data members are **private** here.

Capsule

ENCAPSULATION – EXAMPLE

- **Data Fields are private**
- **Constructors and accessor methods are defined**



ADVANTAGES/BENEFITS OF OOP PROGRAMS

1. Improved software-development productivity
2. Improved software maintainability
3. Faster development
4. Lower cost of development:
5. Higher-quality software:

1. IMPROVED SOFTWARE-DEVELOPMENT PRODUCTIVITY

- Improved software-development productivity: Object-oriented programming is modular, as it provides separation of duties in object-based program development.
- It is also extensible, as objects can be extended to include new attributes and behaviors.
- Objects can also be reused within an across applications. Because of these three factors – modularity, extensibility, and reusability – object-oriented programming provides improved software-development productivity over traditional procedure-based programming techniques.

2. IMPROVED SOFTWARE MAINTAINABILITY

- For the reasons mentioned above, object oriented software is also easier to maintain.
- Since the design is modular, part of the system can be updated in case of issues without a need to make large-scale changes.

3.FASTER DEVELOPMENT:

- Reuse enables faster development.
- Object-oriented programming languages come with rich libraries of objects, and code developed during projects is also reusable in future projects.

4.LOWER COST OF DEVELOPMENT:

- The reuse of software also lowers the cost of development.
- Typically, more effort is put into the object-oriented analysis and design, which lowers the overall cost of development.

5. HIGHER-QUALITY SOFTWARE:

- Faster development of software and lower cost of development allows more time and resources to be used in the verification of the software.
- Although quality is dependent upon the experience of the teams, object oriented programming tends to result in higher-quality software.

→ Next Topic

INTRODUCTION TO JAVA PROGRAMMING